

# CT484: PHÁT TRIỂN ỨNG DỤNG DI ĐỘNG

## XÂY DỰNG ỨNG DỤNG MYSHOP - PHẦN 1

File báo cáo cần nộp là file PDF trong đó có ghi thông tin **mã sinh viên, họ tên, lớp học** phần cùng với **hình minh họa tại các bước kiểm tra kết quả thực thi, các chức năng (không cần chụp hình mã nguồn)**. Cuối file báo cáo ghi đường link đến GitHub mã nguồn của dự án.

Ứng dụng MyShop có các chức năng chính sau:

- Hiển thị và cập nhật danh mục các sản phẩm
- Xem chi tiết một sản phẩm
- Đánh dấu sản phẩm được yêu thích
- Thêm, xóa sản phẩm trong giỏ hàng
- Thực hiện đặt hàng, xem lại đơn hàng, chi tiết đơn hàng
- Đăng ký, đăng nhập
- Lưu trữ dữ liệu trên Firebase

Sinh viên chỉ cần tạo **MỘT** báo cáo duy nhất cho tất cả các buổi thực hành.

### Bước 0: Chuẩn bị môi trường làm việc

- Cài đặt Flutter, xem hướng dẫn tại: <https://docs.flutter.dev/get-started/install>. Cài đầy đủ Android Studio, Android SDK, Android SDK Command-line Tools, và Android SDK Build-Tools.
- Kiểm tra cài đặt môi trường thành công, vào Terminal/Command Prompt, chạy lệnh `flutter doctor`:

```
D:\> flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.0.1, on Microsoft Windows [Version 10.0.22000.708], locale vi-VN)
[✓] Android toolchain - develop for Android devices (Android SDK version 32.1.0-rc1)
[✓] Visual Studio - develop for Windows (Visual Studio Community 2022 17.1.1)
[✓] Android Studio (version 2021.2)
[✓] VS Code (version 1.68.0)
[✓] Connected device (1 available)
[✓] HTTP Host Availability

• No issues found!
```

Kiểm tra rằng **Flutter** và **Android toolchain** không có vấn đề (dấu check xanh).

- Mở *Android Studio* > *Virtual Device Manager* để tạo một máy ảo di động.
- Tải và cài đặt git: <https://git-scm.com/download/win>.
- Kiểm tra rằng có thể gõ lệnh `git` trên Terminal/Command Prompt:

```
D:\> git --version
git version 2.35.2.windows.1
```

(Trong trường hợp không thể chạy lệnh từ Terminal thì cần thêm đường dẫn đến thư mục chứa chương trình `git` vào biến môi trường PATH trên máy của bạn).

- Trình soạn thảo mã nguồn (IDE): có thể dùng Android Studio hoặc Visual Studio Code (VS Code). Nếu sử dụng VS Code thì cần cài đặt thêm phần mở rộng cho [Dart](#) và [Flutter](#).

## Bước 1: Khởi tạo dự án

- Xem video hướng dẫn nhận mã nguồn dự án từ GitHub Classroom, checkout repo về máy:

```
git clone <đường-link-đến-repo-GitHub-đã-nhận>
```

**Chú ý:** Đặt dự án trong thư mục có đường dẫn không chứa khoảng trắng hoặc ký tự tiếng Việt.

- Dùng VS Code mở thư mục dự án, nội dung tập tin *lib/main.dart* như sau:

```
import 'package:flutter/material.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    final colorScheme = ColorScheme.fromSeed(
      seedColor: Colors.purple,
      secondary: Colors.deepOrange,
      background: Colors.white,
      surfaceTint: Colors.grey[200],
    );

    final themData = ThemeData(
      fontFamily: 'Lato',
      colorScheme: colorScheme,
      appBarTheme: AppBarTheme(
        backgroundColor: colorScheme.primary,
        foregroundColor: colorScheme.onPrimary,
        elevation: 4,
        shadowColor: colorScheme.shadow,
      ),
    );

    return MaterialApp(
      title: 'MyShop',
      debugShowCheckedModeBanner: false,
      theme: themData,
      home: Scaffold(
        appBar: AppBar(
          title: const Text('MyShop'),
        ),
        body: const Center(
          child: Text('Welcome to MyShop'),
        ),
      ),
    );
  }
}
```

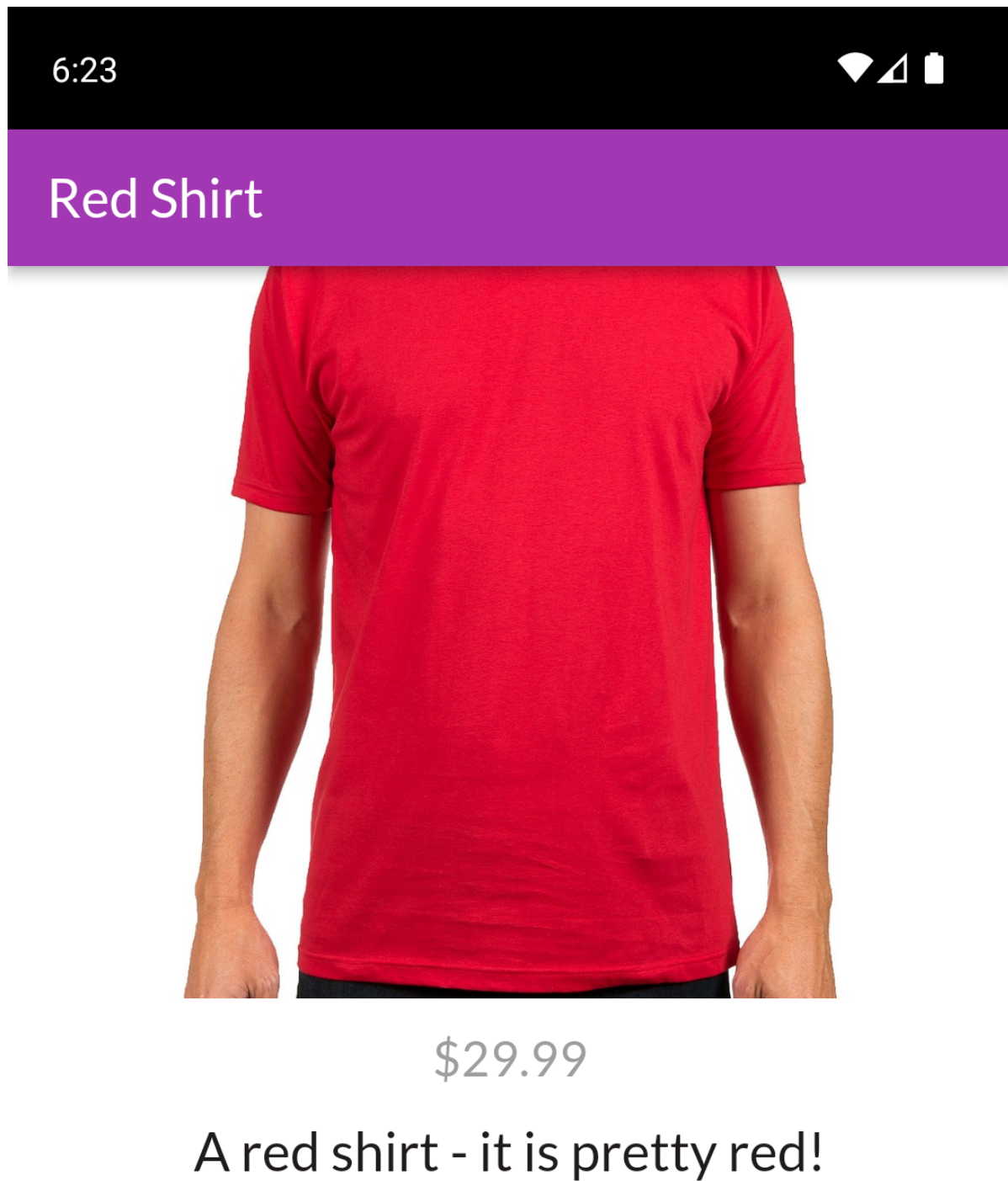
- Sao chép thư mục *assets* đã cho vào thư mục gốc của dự án. Hiệu chỉnh tập tin *pubspec.yaml*, khai báo sử dụng các font trong thư mục *assets*:

```
flutter:
  uses-material-design: true
  fonts:
    - family: Lato
      fonts:
        - asset: assets/fonts/Lato-Regular.ttf
        - asset: assets/fonts/Lato-Bold.ttf
          weight: 700
    - family: Anton
      fonts:
        - asset: assets/fonts/Anton-Regular.ttf
```

- Mở thiết bị giả lập sau đó tiến hành chạy ứng dụng: Run > Run Without Debugging. Kiểm tra ứng dụng chạy thành công.
- Điền các thông tin của bạn vào tập tin *README.md*.
- Lưu các thay đổi vào git và gửi lên GitHub: mở Terminal tại thư mục gốc của dự án và chạy các lệnh sau:

```
git add lib/main.dart pubspec.yaml assets README.md
git commit -m "Đổi theme cho ứng dụng"
git push origin master
```

## Bước 2: Xây dựng trang hiển thị thông tin chi tiết sản phẩm



- Định nghĩa lớp **Product** miêu tả thông tin của một sản phẩm (*lib/models/product.dart*):

```

1  class Product {
2      final String? id;
3      final String title;
4      final String description;
5      final double price;
6      final String imageUrl;
7      final bool isFavorite;
8
9      Product({
10         this.id,
11         required this.title,
12         required this.description,
13         required this.price,
14         required this.imageUrl,
15         this.isFavorite = false,
16     });
17
18     Product copyWith({
19         String? id,
20         String? title,
21         String? description,
22         double? price,
23         String? imageUrl,
24         bool? isFavorite,
25     }) {
26         return Product(
27             id: id ?? this.id,
28             title: title ?? this.title,
29             description: description ?? this.description,
30             price: price ?? this.price,
31             imageUrl: imageUrl ?? this.imageUrl,
32             isFavorite: isFavorite ?? this.isFavorite,
33         );
34     }
35 }
36

```

- Danh sách các sản phẩm ví dụ (*items*) được cho trong tập tin *products.txt*. Định nghĩa lớp **ProductsManager** quản lý các sản phẩm (*lib/ui/products/products\_manager.dart*):

```

1  import '../models/product.dart';
2
3  class ProductsManager {
4  >  final List<Product> _items = [...
40
41    int get itemCount {
42    |   return _items.length;
43    }
44
45    List<Product> get items {
46    |   return [..._items];
47    }
48
49    List<Product> get favoriteItems {
50    |   return _items.where((item) => item.isFavorite).toList();
51    }
52  }

```

- Định nghĩa trang thông tin chi tiết sản phẩm (*lib/ui/products/product\_detail\_screen.dart*):

```

import 'package:flutter/material.dart';

import '../models/product.dart';

class ProductDetailScreen extends StatelessWidget {
  const ProductDetailScreen({
    this.product, {
    super.key,
  });

  final Product product;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(product.title),
      ),
      body: SingleChildScrollView(
        child: Column(
          children: <Widget>[
            SizedBox(
              height: 300,
              width: double.infinity,
              child: Image.network(product.imageUrl, fit: BoxFit.cover),
            ),
            const SizedBox(height: 10),
            Text(
              '\${product.price}',
              style: const TextStyle(color: Colors.grey, fontSize: 20),
            ),
            const SizedBox(height: 10),
            Container(
              padding: const EdgeInsets.symmetric(horizontal: 10),
              width: double.infinity,
              child: Text(
                product.description,
                textAlign: TextAlign.center,
                softWrap: true,
                style: Theme.of(context).textTheme.titleLarge,
              ),
            ),
          ],
        ),
      ),
    );
  }
}

```

Widget [SingleChildScrollView](#) cung cấp một vùng chứa trong đó widget con của nó có thể cuộn được.

- Hiệu chỉnh *lib/main.dart* để kiểm tra trang hiển thị thông tin chi tiết sản phẩm:

```

...
import 'ui/products/products_manager.dart';
import 'ui/products/product_detail_screen.dart';
...

```

```

class MyApp extends StatelessWidget {
  ...
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      ...
      // Đặt ProductDetailScreen làm trang home
      home: SafeArea(
        child: ProductDetailScreen(
          ProductsManager().items[0],
        ),
      ),
    );
  }
}

```

[SafeArea](#) cung cấp cho widget con padding đủ để tránh các thành phần giao diện của hệ điều hành.

- Sau khi kiểm tra, lưu mã nguồn vào repo git và lên GitHub:

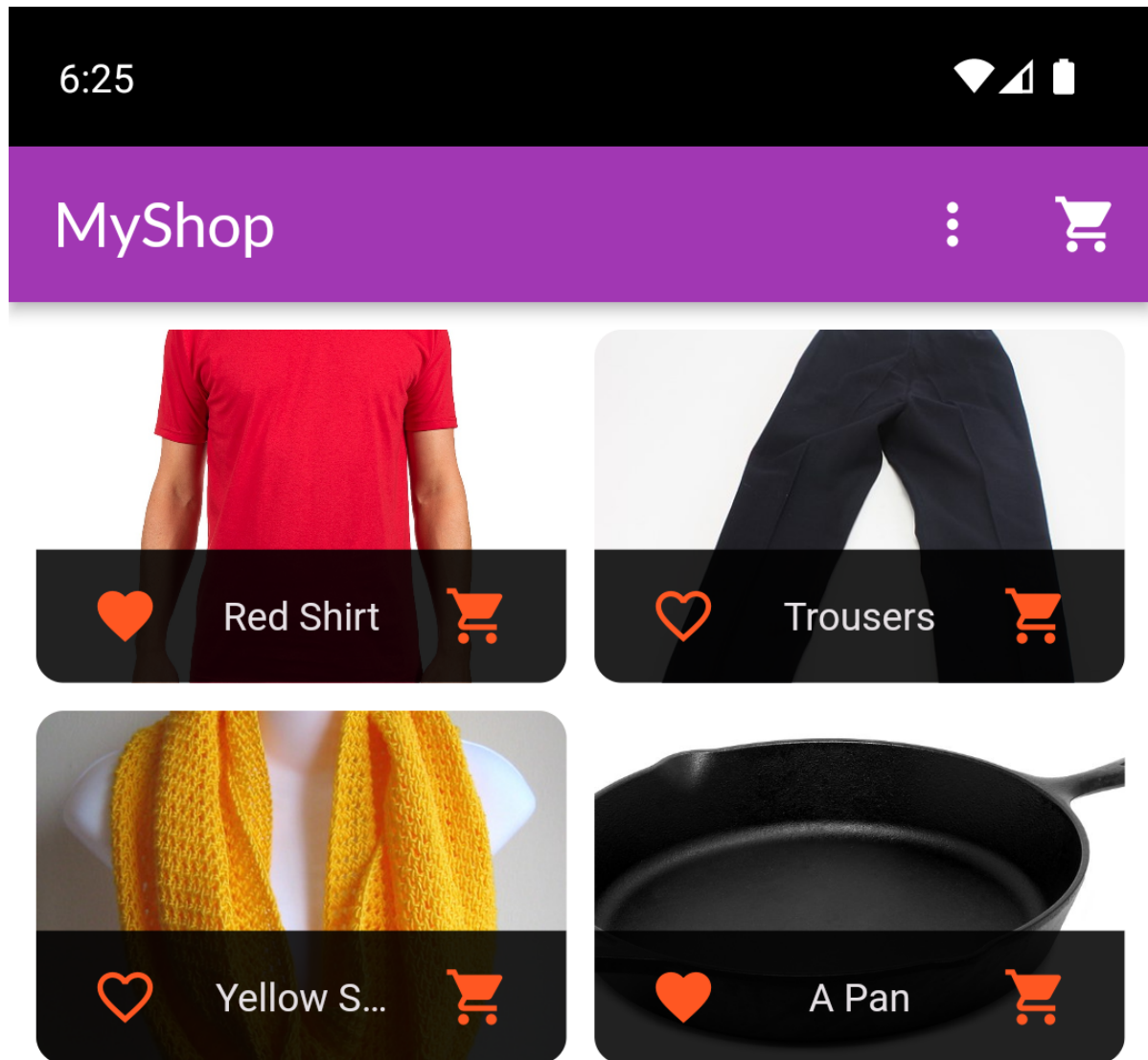
```

git add -u
git add lib/models lib/ui
git commit -m "Xây dựng trang chi tiết sản phẩm"
git push origin master

```



### Bước 3: Xây dựng trang tổng quan các sản phẩm



- Định nghĩa widget **ProductGridTile** trình bày thông tin một sản phẩm (*lib/ui/products/product\_grid\_tile.dart*):

```

import 'package:flutter/material.dart';

import '../models/product.dart';

class ProductGridTile extends StatelessWidget {
  const ProductGridTile(
    this.product, {
    super.key,
  });

  final Product product;

  @override
  Widget build(BuildContext context) {
    return ClipRRect(
      borderRadius: BorderRadius.circular(10),
      child: GridTile(
        footer: ProductGridFooter(),
        child: GestureDetector(
          onTap: () {
            print('Go to product detail screen')
          },
          child: Image.network(
            product.imageUrl,
            fit: BoxFit.cover,
          ),
        ),
      ),
    );
  }
}

class ProductGridFooter extends StatelessWidget {
  const ProductGridFooter({ super.key });

  @override
  Widget build(BuildContext context) {
  }
}

```

```
}
```

Widget *ProductGridFooter* được định nghĩa chi tiết như sau:

```
class ProductGridFooter extends StatelessWidget {
  const ProductGridFooter({
    super.key,
    required this.product,
    this.onFavoritePressed,
    this.onAddToCartPressed,
  });

  final Product product;
  final void Function()? onFavoritePressed;
  final void Function()? onAddToCartPressed;

  @override
  Widget build(BuildContext context) {
    return GridTileBar(
      backgroundColor: Colors.black87,
      leading: IconButton(
        icon: Icon(
          product.isFavorite ? Icons.favorite : Icons.favorite_border,
        ),
        color: Theme.of(context).colorScheme.secondary,
        onPressed: onFavoritePressed,
      ),
      title: Text(
        product.title,
        textAlign: TextAlign.center,
      ),
      trailing: IconButton(
        icon: const Icon(
          Icons.shopping_cart,
        ),
        onPressed: onAddToCartPressed,
        color: Theme.of(context).colorScheme.secondary,
      ),
    );
  }
}
```

Hiệu chỉnh *footer* trong **ProductGridTile** sử dụng *ProductGridFooter* đã định nghĩa:

```
class ProductGridTile extends StatelessWidget {
  ...

  @override
  Widget build(BuildContext context) {
    return ClipRRect(
      borderRadius: BorderRadius.circular(10),
      child: GridTile(
        // Hiệu chỉnh footer
        footer: ProductGridFooter(
          product: product,
```

```

        onFavoritePressed: () {
          print('Toggle a favorite product');
        },
        onAddToCartPressed: () {
          print('Add item to cart');
        },
      ),
      ...
    ),
  );
}

```

- Định nghĩa widget **ProductsGrid** hiển thị các sản phẩm dạng lưới (*lib/ui/products/products\_grid.dart*):

```

import 'package:flutter/material.dart';

import 'product_grid_tile.dart';
import 'products_manager.dart';

class ProductsGrid extends StatelessWidget {
  final bool showFavorites;

  const ProductsGrid({
    super.key,
    required this.showFavorites,
  });

  @override
  Widget build(BuildContext context) {
    final productsManager = ProductsManager();
    final products =
      showFavorites ? productsManager.favoriteItems : productsManager.items;

    return GridView.builder(
      padding: const EdgeInsets.all(10.0),
      itemCount: products.length,
      itemBuilder: (ctx, i) => ProductGridTile(products[i]),
      gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(
        crossAxisCount: 2,
        childAspectRatio: 3 / 2,
        crossAxisSpacing: 10,
        mainAxisSpacing: 10,
      ),
    );
  }
}

```

- Định nghĩa trang tổng quan các sản phẩm (*lib/ui/products/products\_overview\_screen.dart*):

```

import 'package:flutter/material.dart';

import 'products_grid.dart';

enum FilterOptions { favorites, all }

class ProductsOverviewScreen extends StatefulWidget {
  const ProductsOverviewScreen({super.key});

  @override
  State<ProductsOverviewScreen> createState() => _ProductsOverviewScreenState();
}

class _ProductsOverviewScreenState extends State<ProductsOverviewScreen> {
  var _showOnlyFavorites = false;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('MyShop'),
        actions: <Widget>[
          ProductFilterMenu(),
          ShoppingCartButton(),
        ],
      ),
      body: ProductsGrid(
        showFavorites: _showOnlyFavorites,
      ),
    );
  }
}

class ProductFilterMenu extends StatelessWidget {
  const ProductFilterMenu({ super.key });

  @override
  Widget build(BuildContext context) {
  }
}

class ShoppingCartButton extends StatelessWidget {
  const ShoppingCartButton({ super.key });

  @override
  Widget build(BuildContext context) {
  }
}

```

Widget *ProductFilterMenu* và *ShoppingCartIcon* được định nghĩa như sau:

```

class ProductFilterMenu extends StatelessWidget {
  const ProductFilterMenu({super.key, this.onFilterSelected});

  final void Function(FilterOptions selectedValue)? onFilterSelected;

  @override
  Widget build(BuildContext context) {
    return PopupMenuButton(
      onSelected: onFilterSelected,
      icon: const Icon(
        Icons.more_vert,
      ),
      itemBuilder: (ctx) => [
        const PopupMenuItem(
          value: FilterOptions.favorites,
          child: Text('Only Favorites'),
        ),
        const PopupMenuItem(
          value: FilterOptions.all,
          child: Text('Show All'),
        ),
      ],
    );
  }
}

class ShoppingCartButton extends StatelessWidget {
  const ShoppingCartButton({super.key, this.onPressed});

  final void Function()? onPressed;

  @override
  Widget build(BuildContext context) {
    return IconButton(
      icon: const Icon(
        Icons.shopping_cart,
      ),
      onPressed: onPressed,
    );
  }
}

```

Hiệu chỉnh thuộc tính *actions* trong *AppBar* của **ProductsOverScreen**:

```

class _ProductsOverviewScreenState extends State<ProductsOverviewScreen> {
  var _showOnlyFavorites = false;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('MyShop'),
        // Hiệu chỉnh actions
        actions: <Widget>[
          ProductFilterMenu(

```

```

        onFilterSelected: (filter) {
          setState(() {
            if (filter == FilterOptions.favorites) {
              _showOnlyFavorites = true;
            } else {
              _showOnlyFavorites = false;
            }
          });
        },
      ),
      ShoppingCartButton(
        onPressed: () {
          print('Go to cart screen');
        },
      ),
    ],
  ),
  body: ProductsGrid(
    showFavorites: _showOnlyFavorites,
  ),
);
}

```

- Hiệu chỉnh `lib/main.dart` kiểm tra trang tổng quan các sản phẩm:

```

...
import 'ui/products/products_overview_screen.dart';
...
class MyApp extends StatelessWidget {
  ...
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      ...
      // Hiệu chỉnh trang home
      home: const SafeArea(
        child: ProductsOverviewScreen(),
      ),
    );
  }
}

```

- Hiệu chỉnh widget **ProductGridTile** (`lib/ui/products/product_grid_tile.dart`) để thực hiện liên kết đến trang thông tin chi tiết sản phẩm:

```

...
import 'product_detail_screen.dart';

class ProductGridTile extends StatelessWidget {
  ...

  @override
  Widget build(BuildContext context) {
    return ClipRect(

```

```

borderRadius: BorderRadius.circular(10),
child: GridTile(
  ...
  child: GestureDetector(
    onTap: () {
      // Chuyển đến trang ProductDetailScreen
      Navigator.of(context).push(
        MaterialPageRoute(
          builder: (ctx) => ProductDetailScreen(product),
        ),
      );
    },
    child: Image.network(
      product.imageUrl,
      fit: BoxFit.cover,
    ),
  ),
),
);
}
}

```

- Sau khi kiểm tra, lưu mã nguồn vào repo git và lên GitHub:

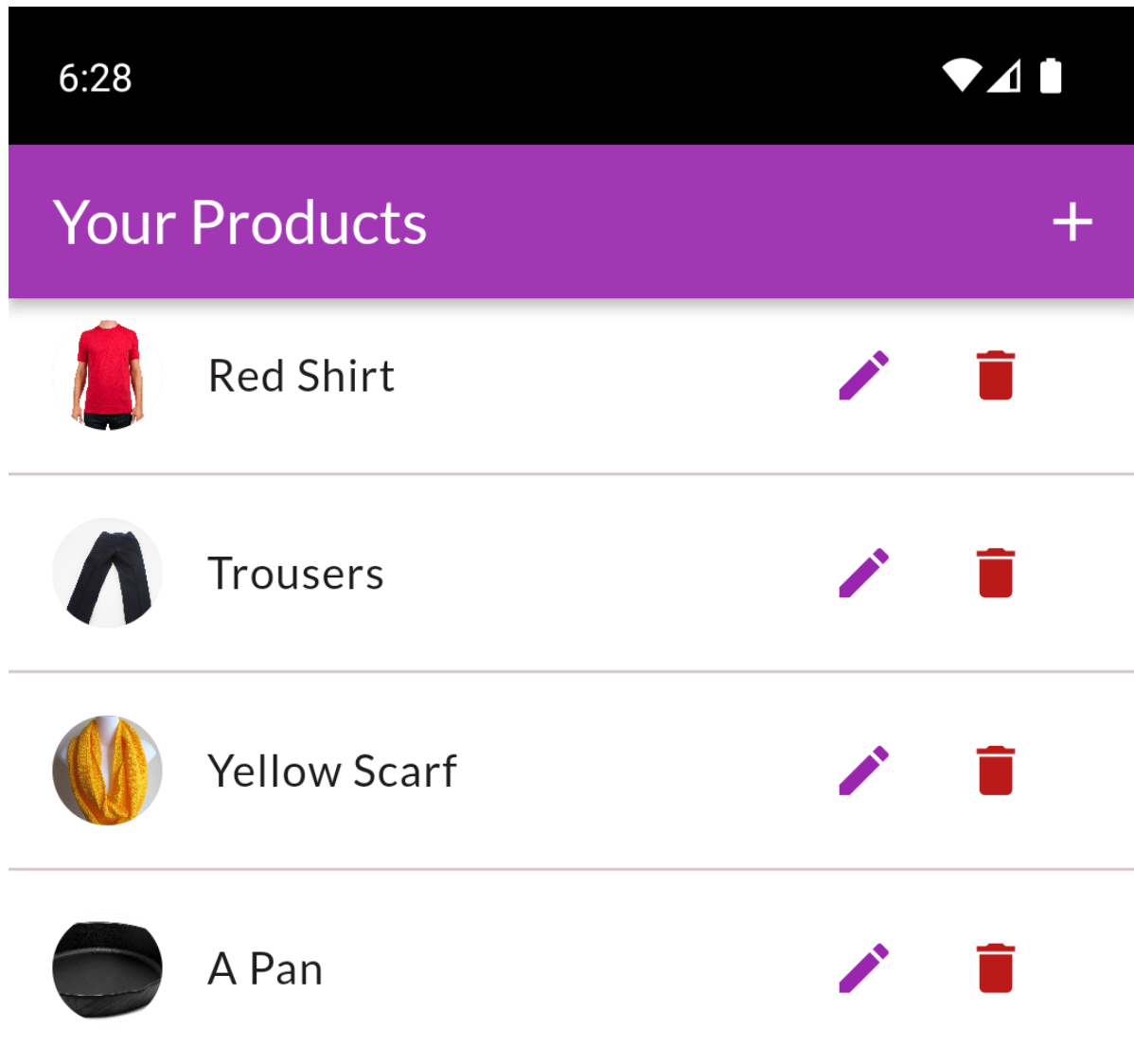
```

git add -u
git add lib/ui
git commit -m "Xây dựng trang tổng quan các sản phẩm"
git push origin master

```



#### Bước 4: Xây dựng trang các sản phẩm của người dùng



- Định nghĩa widget **UserProductListTile** hiển thị thông tin một sản phẩm cùng với các thao tác sửa/xóa (*lib/ui/products/user\_product\_list\_tile.dart*):

```

import 'package:flutter/material.dart';

import '../models/product.dart';

class UserProductListTile extends StatelessWidget {
  final Product product;

  const UserProductListTile(
    this.product, {
    super.key,
  });

  @override
  Widget build(BuildContext context) {
    return ListTile(
      title: Text(product.title),
      leading: CircleAvatar(
        backgroundImage: NetworkImage(product.imageUrl),
      ),
      trailing: SizedBox(
        width: 100,
        child: Row(
          children: <Widget>[
            EditUserProductButton(),
            DeleteUserProductButton(),
          ],
        ),
      ),
    );
  }
}

class DeleteUserProductButton extends StatelessWidget {
  const DeleteUserProductButton({ super.key });

  @override
  Widget build(BuildContext context) {
  }
}

class EditUserProductButton extends StatelessWidget {
  const EditUserProductButton({ super.key });

  @override

```

```
@override  
Widget build(BuildContext context) {  
  
}  
}
```

Widget *DeleteUseProductButton* và *EditUserProductButton* được định nghĩa như sau:

```

class DeleteUserProductButton extends StatelessWidget {
  const DeleteUserProductButton({
    super.key,
    this.onPressed,
  });

  final void Function()? onPressed;

  @override
  Widget build(BuildContext context) {
    return IconButton(
      icon: const Icon(Icons.delete),
      onPressed: onPressed,
      color: Theme.of(context).colorScheme.error,
    );
  }
}

class EditUserProductButton extends StatelessWidget {
  const EditUserProductButton({
    super.key,
    this.onPressed,
  });

  final void Function()? onPressed;

  @override
  Widget build(BuildContext context) {
    return IconButton(
      icon: const Icon(Icons.edit),
      onPressed: onPressed,
      color: Theme.of(context).colorScheme.primary,
    );
  }
}

```

Hiệu chỉnh **UserProductGridTile**:

```

class UserProductListTile extends StatelessWidget {
  ...

  @override
  Widget build(BuildContext context) {
    return ListTile(

```

```

...
trailing: SizedBox(
  width: 100,
  child: Row(
    children: <Widget>[
      // Bắt sự kiện cho nút edit
      EditUserProductButton(
        onPressed: () {
          print('Go to edit product screen');
        },
      ),
      // Bắt sự kiện cho nút delete
      DeleteUserProductButton(
        onPressed: () {
          ScaffoldMessenger.of(context)
            ..hideCurrentSnackBar()
            ..showSnackBar(
              const SnackBar(
                content: Text(
                  'Delete a product',
                  textAlign: TextAlign.center,
                ),
              ),
            ),
        },
      ),
    ],
  ),
);
}

```

- Định nghĩa trang hiển thị các sản phẩm của người dùng (*lib/ui/products/user\_products\_screen.dart*):

```

import 'package:flutter/material.dart';

import 'user_product_list_tile.dart';
import 'products_manager.dart';

class UserProductsScreen extends StatelessWidget {
  const UserProductsScreen({ super.key });

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Your Products'),
        actions: <Widget>[
          AddUserProductButton(),
        ],
      ),
      body: const UserProductList(),
    );
  }
}

class UserProductList extends StatelessWidget {
  const UserProductList({ super.key });

  @override
  Widget build(BuildContext context) {

  }
}

class AddUserProductButton extends StatelessWidget {
  const AddUserProductButton({ super.key });

  @override
  Widget build(BuildContext context) {

  }
}

```

Widget *UserProductList* và *AddUserProductButton* được định nghĩa như sau:

```

class UserProductList extends StatelessWidget {
  const UserProductList({super.key});

  @override
  Widget build(BuildContext context) {
    final productsManager = ProductsManager();

    return ListView.builder(
      itemCount: productsManager.itemCount,
      itemBuilder: (ctx, i) => Column(
        children: [
          UserProductListTile(
            productsManager.items[i],
          ),
          const Divider(),
        ],
      ),
    );
  }
}

class AddUserProductButton extends StatelessWidget {
  const AddUserProductButton({
    super.key,
    this.onPressed,
  });

  final void Function()? onPressed;

  @override
  Widget build(BuildContext context) {
    return IconButton(
      icon: const Icon(Icons.add),
      onPressed: onPressed,
    );
  }
}

```

Hiệu chỉnh **UserProductsScreen**:

```

class UserProductsScreen extends StatelessWidget {

```

```

const UserProductsScreen({super.key});

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Your Products'),
      actions: <Widget>[
        // Bắt sự kiện cho nút add
        AddUserProductButton(
          onPressed: () {
            print('Go to edit product screen');
          },
        ),
      ],
    ),
    body: const UserProductList(),
  );
}

```

- Hiệu chỉnh *lib/main.dart* kiểm tra trang các sản phẩm của người dùng:

```

...
import 'ui/products/user_products_screen.dart';
...
class MyApp extends StatelessWidget {
  ...
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      ...
      // Hiệu chỉnh trang home
      home: const SafeArea(
        child: UserProductsScreen(),
      ),
    );
  }
}

```

- Sau khi kiểm tra, lưu mã nguồn vào repo git và lên GitHub:













```

git add -u
git add lib/ui
git commit -m "Xây dựng trang các sản phẩm người dùng"
git push origin master

```

Cấu trúc thư mục mã nguồn hiện tại như sau:



- ✓  lib
  - ✓  models
    -  product.dart
  - ✓  ui\products
    -  product\_detail\_screen.dart
    -  product\_grid\_tile.dart
    -  products\_grid.dart
    -  products\_manager.dart
    -  products\_overview\_screen.dart
    -  user\_product\_list\_tile.dart
    -  user\_products\_screen.dart
  -  main.dart